Regulating Artificial Inteligence

Joao Guerreiro, Sergio Rebelo, and Pedro Teles

June 2024

Motivation

- Advances in **AI** hold the promise of **large benefits to society**.
- But they also carry the **risk** of significant **societal costs**.
- **Externalities**
  - Promoting political polarization, facilitating fraud, spreading false information, endangering financial stability, weakening democracies.
- **Internalities**
  - Manipulating individuals to act against their self-interest through exploitation of self-control and time-inconsistency problems.

Geoffrey Hinton on the dangers of AI

- "AI will make it much easier for authoritarian governments to manipulate the electorate with fake news that is targeted to each individual."
- AI algorithms "will be able to manipulate people [...] and they will be very good at convincing people 'cause they'll have learned from all the novels that were ever written, all the books by Machiavelli, all the political connivances, they'll know all that stuff. They'll know how to do it."
- "AI may pose more danger than climate change."

Ongoing initiatives

- In May 2023, a group of prominent AI figures declared that "Addressing the existential risks posed by AI should be a global priority, on par with other worldwide challenges like pandemics and nuclear warfare."
- G7 nations launched the Hiroshima AI Process to harmonize AI regulation.
- Europe and the United States have started to design regulatory frameworks.

Ongoing initiatives

California's proposed law SB-1047 requires:

- safety assessments
- Shutdown capability
- Liability for "hazardous capability"

How should AI be regulated?

- A key element of ongoing developments in AI is great uncertainty about the costs and benefits.
  - Users do not consider the costs associated with externalities (or internalities).
- Given this uncertainty, how should AI be regulated?

Related literature

- **AI regulation**: Acemoglu and Lensman (2023), Gans (2024).
- **Impact of AI on the economy**: Burstein, Morales, and Vogel (2019), Jones and Tonetti (2020), Farboodi and Veldkamp (2021), Acemoglu and Restrepo (2022), Jones (2023).
- **Value of experimentation**: Callander (2011), Ilut and Valchev (2023).
- **Design Clinical Trials/Multi-armed Bandits**: Thompson (1933), Gittins (1974).

Outline

1. Baseline model: Externalities + beta testing
   - Unregulated equilibrium.
   - Social optimum.
   - AI regulatory frameworks.
2. Extension I: Heterogeneous expectations
3. Extension II: Imperfect beta testing (probability of obtaining information depends on sample size).
4. Extension III: Internalities.

Baseline Model

Baseline Model

Two periods: $t = 1, 2$.

Agents: households/consumers ($i \in [0, N]$) and one monopolistic developer.

Timing:

1. Developer chooses the level of innovation, ℓ, and pays cost $f(\ell)$.
2. Developer can test/release algorithm at time $t = 1$ – sell to μ_1 users and (imperfectly) evaluate externalities.
3. Developer decides whether to release the algorithm at $t = 2$.
4. At the end of $t = 2$, external costs (both at $t = 1$ and $t = 2$) are known.

Household problem

- Continuum of households $i \in [0, N]$ with preferences

$$\mathcal{U}_i = (1-\beta)\left\{y + [u(\ell)\mu_1 - p_1]\,\mathcal{I}_{1,i} - \mathbb{E}(e_1^2)\right\} + \beta\mathbb{E}\left\{y + [u(\ell)\mu_2 - p_2]\,\mathcal{I}_{2,i} - e_2^2\right\}.$$

- Household purchases AI license in period t ($\mathcal{I}_{i,t} = 1$) as long as

$$p_t \leq u(\ell)\mu_t,$$

$\mathcal{I}_{i,t} = 0$ otherwise.

Externality

$$\mathcal{U}_i = (1-\beta)\left\{y + [u(\ell)\mu_1 - p_1]\,\mathcal{I}_{1,i} - \mathbb{E}(e_1^2)\right\} + \beta\mathbb{E}\left\{y + [u(\ell)\mu_2 - p_2]\,\mathcal{I}_{2,i} - e_2^2\right\}.$$

- AI usage causes negative externality e_t that reduces utility by e_t^2.
- Externality proportional to measure of users:

$$e_t = \phi(\ell) \times \mu_t.$$

- Externality is known at end of period $t = 2$, with new information at beginning of $t = 2$.

Uncertainty about externality

- For each value of ℓ, $\phi(\ell) \in \mathbb{R}$ is a random variable.

$$\mathbb{E}[\phi(\ell)] = 0, \qquad \mathbb{VAR}(\phi(\ell)) = \sigma^2(\ell)$$

- $\sigma^2(\ell)$ increasing and convex in ℓ, and $\sigma(0) = 0$.
  - Uncertainty about external effects is higher the farther the algorithm is from the status quo.

Uncertainty about externality

- For each value of ℓ, $\phi(\ell) \in \mathbb{R}$ is a random variable.

$$\mathbb{E}[\phi(\ell)] = 0, \qquad \mathbb{VAR}(\phi(\ell)) = \sigma^2(\ell)$$

- $\sigma^2(\ell)$ increasing and convex in ℓ, and $\sigma(0) = 0$.
  - Uncertainty about external effects is higher the farther the algorithm is from the status quo.

- **Beta testing**: If $\mu_1 > 0$, then generate signal

$$\mathbb{E}_2[\phi(\ell)] = \hat{\phi}, \qquad \mathbb{VAR}_2(\phi(\ell)) = \hat{\sigma}^2(\ell) < \sigma^2(\ell)$$

  - **Key**: Beta testing cannot resolve all uncertainty [Residual uncertainty]

AI developer problem

$$\mathcal{V} = (1-\beta)\left(\begin{cases}\mu_1 p_1 - \sigma^2(\ell)\mu_1^2, & \text{if } p_1 \le u(\ell)\mu_1 \\ 0, & \text{if } p_1 > u(\ell)\mu_1\end{cases}\right) + \beta\mathbb{E}[\mathcal{V}_2] - f(\ell).$$

$$\mathcal{V}_2 = \begin{cases}\mu_2 p_2 - [\hat{\phi}^2 + \hat{\sigma}^2(\ell)]\mu_2^2, & \text{if } p_2 \le u(\ell)\mu_2 \quad \text{and} \quad \mu_1 > 0, \\ \mu_2 p_2 - \sigma^2(\ell)\mu_2^2, & \text{if } p_2 \le u(\ell)\mu_2 \quad \text{and} \quad \mu_1 = 0, \\ 0, & \text{if } p_2 > u(\ell)\mu_2.\end{cases}$$

Developer:

- Chooses innovation level ℓ.
- Chooses selling price p_t and number of licenses μ_t (test/release).
- Also suffers disutility from the externality.

Developer solution, $t = 2$

- Optimal price $p_2 = u(\ell)\mu_2$.
  - Charge all consumer surplus (perfect price discrimination) – no monopolistic distortion
- Optimal release
  - After beta test, release only if posterior plus residual uncertainty is not too high:

$$\mu_2 = N \text{ if } u(\ell) - (\hat{\phi}^2 + \hat{\sigma}^2(\ell)) \geq 0$$

  - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty

Developer solution, $t = 1$

- Optimal price $p_1 = u(\ell)\mu_1$.
- Optimal testing/release
  - If uncertainty is low, $u(\ell) - \sigma^2(\ell) \geq 0$, release immediately: $\mu_1 = N$
  - If uncertainty is high, $u(\ell) - \sigma^2(\ell) < 0$, beta test the algorithm: $\mu_1 \downarrow 0$.

Efficient allocation

Efficient allocations: Planner's problem

- Utilitarian social welfare: sum of households and developer utilities
  - Transferable utility $\Rightarrow$ only efficiency concerns

$$\mathcal{W} \equiv (1-\beta)\left[Ny + \left\{u(\ell) - (N+1)\sigma^2(\ell)\right\}\mu_1^2\right] + \beta\mathbb{E}[\mathcal{W}_2] - f(\ell).$$

$$\mathcal{W}_2 = \begin{cases} Ny + \left[u(\ell) - (N+1)\left[\hat{\phi}^2 + \hat{\sigma}^2(\ell)\right]\right]\mu_2^2 & \text{if } \mu_1 > 0, \\ Ny + \left[u(\ell) - (N+1)\,\sigma^2(\ell)\right]\mu_2^2 & \text{if } \mu_1 = 0. \end{cases}$$

- Planner: implements efficient/first-best allocation [Social-optimum benchmark]
  - Chooses innovation level ℓ unconstrained.
  - Chooses number of licenses μ_t unconstrained – beta testing if $\mu_1 \in (0, N)$.

Planner solution at $t = 2$

- Optimal release
  - After beta test, release only if posterior plus residual uncertainty is not too high:

$$\mu_2 = N \text{ if } u(\ell) - (N+1)(\hat{\phi}^2 + \hat{\sigma}^2(\ell)) \geq 0$$

  - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty.

Planner solution, $t = 1$

Optimal testing/release

- If uncertainty is low, $u(\ell) - (N+1)\sigma^2(\ell) \geq 0$, release immediately $\mu_1 = N$
- If uncertainty is high, $u(\ell) - (N+1)\sigma^2(\ell) < 0$, beta test the algorithm $\mu_1 \downarrow 0$.

Testing and releasing in the first period

Uncertainty	**Low**	**Moderate**	**High**
$\sigma^2(\ell)$	$\sigma^2(\ell) \leq \frac{u(\ell)}{N+1}$	$\frac{u(\ell)}{N+1} \leq \sigma^2(\ell) \leq u(\ell)$	$\sigma^2(\ell) \geq u(\ell)$
Developer	release	release	test
Planner	release	test	test

- When uncertainty is moderate, developer and planner disagree.
  - Developer has higher incentives to release/lower incentives to beta test.

Release decisions, $t = 2$

Externality	**Low**	**Moderate**	**High**
$\psi(\ell)^2$	$\psi(\ell)^2 \leq \frac{u(\ell)}{N+1}$	$\frac{u(\ell)}{N+1} \leq \psi(\ell)^2 \leq u(\ell)$	$\psi(\ell)^2 \geq u(\ell)$
Developer	release	release	not release
Planner	release	not release	not release

$$\psi(\ell)^2 \equiv \hat{\phi}^2 + \hat{\sigma}^2(\ell)$$

- When uncertainty is moderate, developer and planner disagree.
  - Developer has higher incentives to release.

Unregulated equilibrium vs. efficient allocation

- Relative to unregulated equilibrium the efficient allocation requires:
  1. Containment of risks associated with AI release
  2. Proper allocation of resources
- Planner can be more cautious in two ways:
  1. More beta testing, and more often withdraws the product in the second period.
  2. Choose a lower, less risky innovation level ℓ.
- Interestingly, the efficient allocation may feature a higher innovation level, ℓ, than the unregulated equilibrium.
  - Since more caution in testing and implementing.

Regulation

Regulating AI - unlimited liability

- Unregulated equilibrium does not deliver efficient allocation.

Q: Are there policies that can recover efficiency?

- **Standard answer**: with externalities, Pigouvian taxes can be used to restore efficiency.
- Here, Pigouvian taxes could take the form of ex-post **liability** for damages caused.

Regulating AI - unlimited liability

Proposition

Suppose the regulator charges the developer $\tau_t = N \times e_t^2 = N \times \phi(\ell)^2 \mu_t^2$. Then, private and social incentives are aligned, and the regulated equilibrium allocations coincide with the efficient allocations.

Regulating AI - limited liability

- But, what if the damages are too large?
  - What if they are higher than the developer's profits?

Regulating AI - limited liability

- But, what if the damages are too large?
  - What if they are higher than the developer's profits?
- In practice, developers are protected by **limited liability**:

$$\tau_t = \min\{N\phi(\ell), u(\ell)\}\mu_t^2$$

- Test/Release incentives are no longer aligned. Why?
  - Since their losses are limited, developers have an incentive to gamble by releasing moderately risky algorithms.

Regulating AI - limited liability

- But, what if the damages are too large?
  - What if they are higher than the developer's profits?
- In practice, developers are protected by **limited liability**:

$$\tau_t = \min\{N\phi(\ell), u(\ell)\}\mu_t^2$$

- Test/Release incentives are no longer aligned. Why?
  - Since their losses are limited, developers have an incentive to gamble by releasing moderately risky algorithms.

Proposition

If developers are protected by limited liability, then (1) they may forego beta testing potentially socially harmful algorithms, and (2) they may release socially harmful algorithms in the second period.

- Conclusion: this policy fails to contain risks associated with more innovative algorithms.

Regulating AI - ex-ante taxes

- We have assumed taxes condition on ex-post realized damages $\tau_t = N \times e_t^2$.
- But, what if we can condition taxes on ex-ante beliefs?

Regulating AI - ex-ante taxes

Proposition

Suppose the regulator mandates a payment equal to $\tau_t^{ex-ante} = N \times \mathbb{E}_t[e_t^2]$*:*

$$\tau_1^{ex-ante} = N\sigma^2(\ell)\mu_1^2 \quad \textit{and} \quad \tau_2^{ex-ante} = N[\hat{\phi}^2 + \hat{\sigma}^2(\ell)]\mu_2^2.$$

Then, private and social incentives are aligned, and the regulated equilibrium delivers the efficient allocation.

- This policy can equivalently be implemented as a sales tax.

Regulating AI - ex-ante taxes with limited liability

Even with limited liability, these taxes deliver the efficient allocation. Why?

- Whenever limited liability binds, both the developer and regulator agree not to sell any AI licenses.

Corollary

Ex-ante taxes deliver the efficient allocation even under the presence of limited liability.

But, there is another issue with taxes that condition on beliefs.

- What if developers have different beliefs than the regulator?

Extension I: Heterogeneous expectations

Heterogeneous expectations

- What if developers and the rest of society disagree about the potential social damages?
- Each holds their beliefs:
  - Societal beliefs

$$\mathbb{VAR}^s[\phi(\ell)^2] = \sigma_s^2(\ell), \qquad \mathbb{E}_2^s[\phi(\ell)] = \hat{\phi}_s, \quad \mathbb{VAR}_2^s[\phi(\ell)] = \hat{\sigma}_s^2(\ell)$$

  - Developer beliefs

$$\mathbb{VAR}^d[\phi(\ell)^2] = \sigma_d^2(\ell), \qquad \mathbb{E}_2^d[\phi(\ell)] = \hat{\phi}_d, \quad \mathbb{VAR}_2^d[\phi(\ell)] = \hat{\sigma}_d^2(\ell)$$

- Focus on the case in which developers are more optimistic: predicts lower variance and lower externality conditional on the same signal.

Heterogeneous expectations - regulation

- With disagreement, ex-post liability policies fail to deliver the efficient allocation
  - Overly optimistic developers still believe they are relatively unlikely to pay for these harms.
    - $\hat{\phi}_d^2 + \hat{\sigma}_d^2(\ell) < \hat{\phi}_s^2 + \hat{\sigma}_s^2(\ell) \Rightarrow$ the developer may release in $t = 2$ when it is not optimal.
    - $\sigma_d^2(\ell) < \sigma_s^2(\ell) \Rightarrow$ the developer may not beta test when it is not socially optimal.

Heterogeneous expectations - regulation

- With disagreement, ex-post liability policies fail to deliver the efficient allocation
  - Overly optimistic developers still believe they are relatively unlikely to pay for these harms.
    - $\hat{\phi}_d^2 + \hat{\sigma}_d^2(\ell) < \hat{\phi}_s^2 + \hat{\sigma}_s^2(\ell) \Rightarrow$ the developer may release in $t = 2$ when it is not optimal.
    - $\sigma_d^2(\ell) < \sigma_s^2(\ell) \Rightarrow$ the developer may not beta test when it is not socially optimal.
- What about ex-ante liability?
  - Whose beliefs to use to design the policy?

Heterogeneous expectations - regulation

- With disagreement, ex-post liability policies fail to deliver the efficient allocation
  - Overly optimistic developers still believe they are relatively unlikely to pay for these harms.
    - $\hat{\phi}_d^2 + \hat{\sigma}_d^2(\ell) < \hat{\phi}_s^2 + \hat{\sigma}_s^2(\ell) \Rightarrow$ the developer may release in $t = 2$ when it is not optimal.
    - $\sigma_d^2(\ell) < \sigma_s^2(\ell) \Rightarrow$ the developer may not beta test when it is not socially optimal.
- What about ex-ante liability?
  - Whose beliefs to use to design the policy?
  - Regulator must design a policy to (1) induce developers to internalize the externality, and (2) correct differences in beliefs

Heterogeneous expectations - regulation

- With disagreement, ex-post liability policies fail to deliver the efficient allocation
  - Overly optimistic developers still believe they are relatively unlikely to pay for these harms.
    - $\hat{\phi}_d^2 + \hat{\sigma}_d^2(\ell) < \hat{\phi}_s^2 + \hat{\sigma}_s^2(\ell) \Rightarrow$ the developer may release in $t = 2$ when it is not optimal.
    - $\sigma_d^2(\ell) < \sigma_s^2(\ell) \Rightarrow$ the developer may not beta test when it is not socially optimal.
- What about ex-ante liability?
  - Whose beliefs to use to design the policy?
  - Regulator must design a policy to (1) induce developers to internalize the externality, and (2) correct differences in beliefs
  - But, that requires knowledge of developers' beliefs...
  - Do regulators have information on these? Can they elicit those beliefs?
    - Developers may pretend to be pessimistic to receive a subsidy from the planner.
- Conclusion: liability policies are unlikely to contain risks associated with AI release.

Beta testing and regulatory approval

Beta testing and regulatory approval

- Is there an alternative policy that can at least contain risks?
- The regulator can **mandate beta testing and regulate approval**. [As in medical trials]

Beta testing and regulatory approval

- Does this policy implement the efficient allocation?

Proposition

Suppose that the regulator mandates beta testing and regulates approval, then:

1. *Only expected socially beneficial algorithms are released both at $t = 1$ and $t = 2$.*

Beta testing and regulatory approval

- Does this policy implement the efficient allocation?

Proposition

Suppose that the regulator mandates beta testing and regulates approval, then:

1. *Only expected socially beneficial algorithms are released both at $t = 1$ and $t = 2$.*
2. *However, the developer chooses a higher innovation level than socially efficient.*

- Risks are contained, but the developer still has an incentive to gamble on a riskier AI.
  - Wasteful spending of resources relative to efficient allocation

But, can this policy improve welfare?

- Beta testing and regulatory approval fail to deliver the efficient allocation.
- Implementing the efficient allocation requires:
  1. Containment of risks associated with AI release ✓
  2. Proper allocation of resources ✗
- But, can this policy at least ensure a welfare improvement relative to forbidding AIs?

But, can this policy improve welfare?

- Beta testing and regulatory approval fail to deliver the efficient allocation.
- Implementing the efficient allocation requires:
  1. Containment of risks associated with AI release ✓
  2. Proper allocation of resources ✗
- But, can this policy at least ensure a welfare improvement relative to forbidding AIs?

Corollary

If the risks are smaller than the expected profits, $(N+1)\sigma^2(\ell)N^2 \leq \Pi(\ell)$*, then the production of the AI is welfare enhancing.*

Conclusions

Conclusion

- Standard policy (Pigouvian taxes) prescriptions fail to deliver efficiency if:
  - Developers are protected by limited liability
  - Developers are relatively more optimistic than the rest of society

- Mandating beta testing and regulating approval:
  - Mitigates risks associated with AI release
  - May lead to over-expenditure of resources in AI development compared to the efficient allocations
  - Nevertheless, it could still improve welfare compared to not developing any algorithm.

EU Artificial Intelligence Act (updated May 30, 2024)

1. Classify AI according to risk
   - unacceptable risks are prohibited,
   - high-risk AIs are regulated,
   - limited risk has lighter transparency regulation,
   - minimal risk is unregulated.
2. Obligations fall on AI developers.
3. Users have fewer obligations.
4. General purpose AI (GPAI):
   - Providers must provide technical documentation, instructions for use, comply with Copyright Directive, and publish summary of content used for training.
   - Free and open licenses only need to comply with copyright and publish training data, unless systemic risk.
   - With systemic risk, must also provide **model evaluations, adversarial testing, track and report serious incidents and ensure cybersecurity protections.**

Thank you!

Other instruments

What else? (but no time today)

Also discuss two other issues:

1. What other instruments/uses of available instruments? More
2. Imperfect-beta testing: what if beta testing requires a large enough mass to generate results? More
3. Internalities: manipulating individuals to act against their self-interest – misinformation or exploitation of self-control/time-inconsistency problems More

Dictatorial regulatory approval

One solution could be to forbid all AIs other than the efficient ℓ^*

- Then, still mandate beta testing and regulatory approval.

Problem: this solution requires a commitment not to allow commercialization of licenses other than the one that maximizes welfare

Dictatorial regulatory approval

One solution could be to forbid all AIs other than the efficient ℓ^*

- Then, still mandate beta testing and regulatory approval.

Problem: this solution requires a commitment not to allow commercialization of licenses other than the one that maximizes welfare

Proposition

Given ℓ, the only time consistent regulatory policy is:

- *Mandate beta testing if $\frac{u(\ell)}{N+1} \leq \sigma_s^2(\ell)$ and immediately release otherwise.*
- *Allow commercialization in time 2 if and only if $\hat{\phi}_s^2 + \hat{\sigma}_s^2(\ell) \leq \frac{u(\ell)}{N+1}$.*

Extension II: Imperfect beta testing

Imperfect beta testing

Back

- Previous model assumes *perfect* beta testing:
  - Testing infinitesimal samples ($\mu_1 \downarrow 0$) resolves uncertainty.
  - Beta testing fully reveals $\phi(\ell)$ – no residual uncertainty.
- Likely unrealistic
  - Informative beta testing may require sizeable samples.
  - Even if successful, beta testing may never be fully informative
- Extension: beta testing succeeds with probability $\pi(\mu_1) = \max\{(\mu_1/\kappa)^\alpha, 1\}$
  - $\alpha > 0$ controls how success probability increases with μ_1.
  - $\kappa \in (0, N]$ is smallest sample size that guarantees success.
  - If test is successful, posterior beliefs are

$$\mathbb{E}_2[\phi(\ell)] = \phi_p(\ell), \qquad Var_2(\phi(\ell)) = \sigma_p^2(\ell) < \sigma^2(\ell)$$

  - If the test is unsuccessful, priors are unchanged.
  - Nests previous case: $\alpha \to 0$ and $\sigma_p^2(\ell) = 0$.

Unregulated equilibrium

Back

- Household problem is the same, buy if $p_t \leq u(\ell)\mu_t$
- Developer's problem

$$\mathcal{V} = (1-\beta)\left(\begin{cases}\mu_1 p_1 - \sigma^2(\ell)\mu_1^2, & \text{if } p_1 \leq u(\ell)\mu_1 \\ 0, & \text{if } p_1 > u(\ell)\mu_1\end{cases}\right) + \beta\mathbb{E}[\mathcal{V}_2] - f(\ell).$$

$$\mathcal{V}_2 = \begin{cases}\mu_2 p_2 - (\phi_p(\ell)^2 + \sigma_p^2(\ell))\mu_2^2, & \text{if } p_2 \leq u(\ell)\mu_2 \quad \text{and beta test successful,} \\ \mu_2 p_2 - \sigma^2(\ell)\mu_2^2, & \text{if } p_2 \leq u(\ell)\mu_2 \quad \text{and beta test unsuccessful,} \\ 0, & \text{if } p_2 > u(\ell)\mu_2.\end{cases}$$

Developer solution at $t = 2$

Back

- Optimal price $p_2 = u(\ell)\mu_2$.
- Optimal release
  - After successful beta test, release only if posterior plus residual uncertainty is not too high: $\mu_2 = N$ if
$$u(\ell) - (\phi_p(\ell)^2 + \sigma_p^2(\ell)) \geq 0$$
    - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty
  - After unsuccessful beta test: algorithm is always withdrawn from the market.

Developer solution at $t = 2$

Back

- Optimal price $p_2 = u(\ell)\mu_2$.
- Optimal release
  - After successful beta test, release only if posterior plus residual uncertainty is not too high: $\mu_2 = N$ if
$$u(\ell) - (\phi_p(\ell)^2 + \sigma_p^2(\ell)) \geq 0$$
    - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty
  - After unsuccessful beta test: algorithm is always withdrawn from the market.

Lemma

The developer's expected utility at $t = 2$ is increasing in μ_1

Developer solution at $t = 1$

Back

- Optimal price $p_1 = u(\ell)\mu_1$.
- Optimal testing/release
  - If uncertainty is low, $u(\ell) - \sigma^2(\ell) \geq 0$, release immediately $\mu_1 = N$
  - If uncertainty is high, $u(\ell) - \sigma^2(\ell) < 0$, beta test the algorithms:

[Assume $\alpha < 2$]

$$\mu_1 = \min\left\{\left[\frac{\alpha}{2}\Lambda^d(\ell)\right]^{\frac{1}{2-\alpha}}, 1\right\}\kappa$$

$$\pi(\mu_1) = \min\left\{\left[\frac{\alpha}{2}\Lambda^d(\ell)\right]^{\frac{\alpha}{2-\alpha}}, 1\right\}$$

Developer's information benefit-cost ratio

$$\Lambda^d(\ell) \equiv \frac{\beta}{1-\beta}\frac{\mathbb{E}[\max\{u(\ell) - (\phi_p(\ell)^2 + \sigma_p^2(\ell)), 0\}]}{\sigma^2(\ell) - u(\ell)}$$

Planner's problem

Back

- Utilitarian social welfare: sum of households and developer utilities
  - Transferable utility $\Rightarrow$ only efficiency concerns

$$\mathcal{W} \equiv (1-\beta)\left[Ny + \left\{u(\ell) - (N+1)\sigma^2(\ell)\right\}\mu_1^2\right] + \beta\mathbb{E}[\mathcal{W}_2] - f(\ell).$$

$$\mathcal{W}_2 = \begin{cases} Ny + \left[u(\ell) - (N+1)\left(\phi_p(\ell)^2 + \sigma_p^2(\ell)\right)\right]\mu_2^2 & \text{if beta test successful,} \\ Ny + \left[u(\ell) - (N+1)\sigma^2(\ell)\right]\mu_2^2 & \text{if beta test unsuccessful.} \end{cases}$$

- Planner: implements efficient/first-best allocation [Social-optimum benchmark]
  - Chooses number of licenses μ_t unconstrained – beta testing if $\mu_1 \in (0, N)$.
  - Chooses innovation level ℓ unconstrained.

Planner solution at $t = 2$

Back

- Optimal release
  - After successful beta test, release only if posterior plus residual uncertainty is not too high: $\mu_2 = N$ if

$$u(\ell) - (N+1)(\phi_p(\ell)^2 + \sigma_p^2(\ell)) \geq 0$$

    - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty.
  - After unsuccessful beta test: algorithm is always withdrawn from the market.
    - Uncertainty is too high and beta testing did not reduce it.

Planner solution at $t = 2$

Back

- Optimal release
  - After successful beta test, release only if posterior plus residual uncertainty is not too high: $\mu_2 = N$ if
$$u(\ell) - (N+1)(\phi_p(\ell)^2 + \sigma_p^2(\ell)) \geq 0$$
    - Some algorithms are withdrawn, even if point belief is not too bad – residual uncertainty.
  - After unsuccessful beta test: algorithm is always withdrawn from the market.
    - Uncertainty is too high and beta testing did not reduce it.

Lemma

The expected social welfare at $t = 2$ is increasing in μ_1.

Planner solution at $t = 1$

Back

- Optimal testing/release
  - If uncertainty is low, $u(\ell) - (N+1)\sigma^2(\ell) \geq 0$, release immediately $\mu_1 = N$
  - If uncertainty is high, $u(\ell) - (N+1)\sigma^2(\ell) < 0$, test the algorithms:

[Assume $\alpha < 2$]

$$\mu_1 = \min\left\{\left[\frac{\alpha}{2}\Lambda^P(\ell)\right]^{\frac{1}{2-\alpha}}, 1\right\}\kappa$$

$$\pi(\mu_1) = \min\left\{\left[\frac{\alpha}{2}\Lambda^P(\ell)\right]^{\frac{\alpha}{2-\alpha}}, 1\right\}$$

Planner's information benefit-cost ratio

$$\Lambda^d(\ell) \equiv \frac{\beta}{1-\beta}\frac{\mathbb{E}[\max\{u(\ell) - (N+1)(\phi_p(\ell)^2 + \sigma_p^2(\ell)), 0\}]}{(N+1)\sigma^2(\ell) - u(\ell)}$$

Testing and releasing in the first period

Back

Uncertainty	Low	Medium	High
$\sigma^2(\ell)$	$\sigma^2(\ell) \leq \frac{u(\ell)}{N+1}$	$\frac{u(\ell)}{N+1} < \sigma^2(\ell) \leq u(\ell)$	$\sigma^2(\ell) > u(\ell)$
Developer	Release	Release	Test $\mu_1 = \min\left\{\left[\frac{\alpha}{2}\Lambda^d(\ell)\right]^{\frac{1}{2-\alpha}}, 1\right\}\kappa$
Planner	Release	Test $\mu_1 = \min\left\{\left[\frac{\alpha}{2}\Lambda^p(\ell)\right]^{\frac{1}{2-\alpha}}, 1\right\}\kappa$	Test $\mu_1 = \min\left\{\left[\frac{\alpha}{2}\Lambda^p(\ell)\right]^{\frac{1}{2-\alpha}}, 1\right\}\kappa$

- Planner more cautious – tests more often and uses smaller samples $\Lambda^p(\ell) < \Lambda^d(\ell)$.

Releasing in the second period

Back

Posterior externality	Low	Medium	High
$\psi_p(\ell)$	$\psi_p(\ell) \leq \frac{u(\ell)}{N+1}$	$\frac{u(\ell)}{N+1} \leq \psi_p(\ell) \leq u(\ell)$	$\psi_p(\ell) \geq u(\ell)$
Developer	Release	Release	Not Release
Planner	Release	Not Release	Not Release

$$\psi_p(\ell) \equiv \begin{cases} \phi_p(\ell)^2 + \sigma_p^2(\ell) & \text{if beta test successful} \\ \sigma^2(\ell) & \text{if beta test unsuccessful} \end{cases}$$

Optimality of different types of regulation

Back

Testing	Decision	No Liability	Limited Liability	Unlimited Liability
Chosen by developer	ℓ	suboptimal	suboptimal	optimal
	$t = 1$ Testing/release	suboptimal	suboptimal	optimal
	$t = 2$ Release	suboptimal	optimal	optimal
Mandatory	ℓ	suboptimal	suboptimal	optimal†
	$t = 1$ Testing/release	optimal†	optimal†	optimal†
	$t = 2$ Release	suboptimal	suboptimal	optimal

† when testing is socially optimal.

Analysis of regulatory frameworks

Residual uncertainty $\Rightarrow$ only unlimited liability achieves social optimum.

- Very different from simple model. Why?

With limited liability:

- Developers still have an incentive to gamble with risky releases at time $t = 2$.
- This incentive also influences their choise of innovation level – innovation can be larger or smaller than social optimum.
- Mandating beta testing is not sufficient because beta tests cannot fully remove uncertainty.

Analysis of regulatory frameworks

Residual uncertainty $\Rightarrow$ only unlimited liability achieves social optimum.

- Very different from simple model. Why?

With limited liability:

- Developers still have an incentive to gamble with risky releases at time $t = 2$.
- This incentive also influences their choise of innovation level – innovation can be larger or smaller than social optimum.
- Mandating beta testing is not sufficient because beta tests cannot fully remove uncertainty.

Q: Is there a policy that ensures new algorithm at least improves welfare?

Do different regulations ensure new algorithm improves welfare?

Back

- Limited liability: does not ensure welfare improvement
  - Developers may release socially harmful algorithms
- Controlled beta test and approval:

Do different regulations ensure new algorithm improves welfare?

Back

Testing	Decision	No Liability	Limited Liability	Unlimited Liability
Chosen by developer	ℓ	suboptimal	suboptimal	optimal
	$t = 1$ Testing/release	suboptimal	suboptimal	optimal
	$t = 2$ Release	suboptimal	optimal	optimal
Chosen by regulator	ℓ	suboptimal	suboptimal	optimal†
	$t = 1$ Testing/release	optimal†	optimal†	optimal†
	$t = 2$ Release	suboptimal	suboptimal	optimal

† when testing is socially optimal.

Extension III: Internalities

Internalities

- Decision utility:

$$\mathcal{U}_i^s \equiv (1-\beta)\left\{y + [u(\ell) - p_1] \times \mathcal{I}_{1,i}\right\} + \beta \mathbb{E}\left[y + [u(\ell) - p_2] \times \mathcal{I}_{2,i}\right].$$

- Experience utility:

$$\begin{aligned}\mathcal{U}_i \equiv (1-\beta)&\left\{y + [u(\ell) - p_1] \times \mathcal{I}_{1,i} - \mathbb{E}[\phi(\ell)^2] \times \mathcal{I}_{1,i}\right\} \\ &+ \beta \mathbb{E}\left[y + [u(\ell) - p_2] \times \mathcal{I}_{2,i} - \phi(\ell)^2 \times \mathcal{I}_{2,i}\right],\end{aligned}$$

Developer's utility

Back

Developer's utility in the second period:

$$\mathcal{V}_2 = \begin{cases} \mu_2 p_2 & \text{if } p \leq u(\ell), \\ 0 & \text{if } p > u(\ell). \end{cases}$$

Maximized utility in the second period

$$\mathcal{V}_2^*(\ell) = N u(\ell).$$

Developer's utility in the first period

$$\mathcal{V} = (1-\beta)\left(\begin{cases} \mu_1 p_1, & \text{if } p_1 \leq u(\ell) \\ 0, & \text{if } p_1 > u(\ell) \end{cases}\right) + \beta \mathbb{E}[\mathcal{V}_2^*(\ell)] - f(\ell).$$

Testing or releasing in the first period, model with internalities

Back

Uncertainty	Low	High
$\sigma^2(\ell)$	$\sigma^2(\ell) \leq u(\ell)$	$\sigma^2(\ell) > u(\ell)$
Developer	release	release
Planner	release	test

Release decisions in the second period, model with internalities

Back

Uncertainty	**Low**	**High**
$\phi(\ell)^2$	$\phi(\ell)^2 \leq u(\ell)$	$\phi(\ell)^2 > u(\ell)$
Developer	release	release
Planner	release	not release

Optimality of different types of regulation, model with internalities

Back

Testing choice	Decision	No Liability	Limited Liability	Unlimited Liability
Developer	ℓ	suboptimal	suboptimal	optimal
	$t = 1$ Testing	suboptimal	suboptimal	optimal
	$t = 2$ Release	suboptimal	optimal	optimal
Regulator	ℓ	$\ell^{\text{eq}} > \ell^*$	optimal†	optimal
	$t = 1$ Testing	suboptimal	optimal	optimal
	$t = 2$ Release	suboptimal	optimal	optimal

† when testing is socially optimal.

Internalities versus externalities

Back

- In the model with externalities, the developer overlooks external impacts on the population but personally experiences these effects. These external effects increase with the number of algorithm users.
- When externalities are high, the developer is dissuaded from releasing the algorithm.
- This restraining factor is absent in the model with internalities.
- This follows from a natural assumption: the developer is not affected by internalities in its own choices, because
  - it does not use the AI algorithm or
  - it is more sophisticated than the households.